sensors

MDPI

Article

A Multilane Tracking Algorithm Using IPDA with Intensity Feature

Behzad Akbari [1,*], Jeyan Thiyagalingam [2], Richard Lee [3] and Thia Kirubarajan [1]

[1] ECE Department, McMaster University, Hamilton, ON L8S 4L8, Canada; kiruba@mcmaster.ca
[2] Rutherford Appleton Laboratory, Scientific Computing Department, Science and Technology Facilities Council, Didcot OX11 0FA, UK; t.jeyan@stfc.ac.uk
[3] General Dynamics Land Systems—Canada, London, ON L8S 4L8, Canada; leer2@gdls.com
* Correspondence: akbarib@mcmaster.ca

Abstract: Detection of multiple lane markings on road surfaces is an important aspect of autonomous vehicles. Although a number of approaches have been proposed to detect lanes, detecting multiple lane markings, particularly across a large number of frames and under varying lighting conditions, in a consistent manner is still a challenging problem. In this paper, we propose a novel approach for detecting multiple lanes across a large number of frames and under various lighting conditions. Instead of resorting to the conventional approach of processing each frame to detect lanes, we treat the overall problem as a multitarget tracking problem across space and time using the integrated probabilistic data association filter (IPDAF) as our basis filter. We use the intensity of the pixels as an augmented feature to correctly group multiple lane markings using the Hough transform. By representing these extracted lane markings as splines, we then identify a set of control points, which becomes a set of targets to be tracked over a period of time, and thus across a large number of frames. We evaluate our approach on two different fronts, covering both model- and machine-learning-based approaches, using two different datasets, namely the Caltech and TuSimple lane detection datasets, respectively. When tested against model-based approach, the proposed approach can offer as much as 5%, 12%, and 3% improvements on the true positive, false positive, and false positives per frame rates compared to the best alternative approach, respectively. When compared against a state-of-the-art machine learning technique, particularly against a supervised learning method, the proposed approach offers 57%, 31%, 4%, and 9× improvements on the false positive, false negative, accuracy, and frame rates. Furthemore, the proposed approach retains the explainability, or in other words, the cause of actions of the proposed approach can easily be understood or explained.

Keywords: multilane tracking; probability density function (PDF); maximum a posteriori (MAP); integrated probability data association (IPDA); curve fitting; Hough transform

Citation: Akbari, B.; Thiyagalingam, J.; Lee, R.; Kirubarajan, T. A Multilane Tracking Algorithm Using IPDA with Intensity Feature. *Sensors* **2021**, *21*, 461. https://doi.org/10.3390/s21020461

Received: 3 December 2020
Accepted: 7 January 2021
Published: 11 January 2021

1. Introduction

Advanced driving assistance systems (ADAS) are no longer an optional or a luxurious component in modern vehicles [1,2]. Instead, they are becoming a core component, especially with the migration towards autonomous vehicles. ADAS covers a number of varying functionalities, such as lane departure warning (LDW), lane keep assist (LKA), lane change merge (LCM), adaptive cruise control (ACC), collision detection and avoidance (CD), night vision, and blind spot detection, to mention a few [1,3–13]. The overall functionality of the ADAS is underpinned by a machine vision component whose ability to understand the surroundings, particularly the ability to extract lane boundaries and markings in roads. With ADAS becoming a core component, it is essential that potential errors arising out of the machine vision component be as low as possible. However, correctly, consistently, and constantly extracting lane markings across a range of weather conditions is not trivial. In addition to this, varying lane marking standards, obscure lane markings, splitting

and merging of lanes, and shadows of vehicles and objects exacerbate this problem even more [14–17]. We show a number of such examples in Figure 1.

Figure 1. Example cases where extracting lane markings is challenging (**A**) patched road; (**B**) Shadow; (**C**) cross walk; (**D**) road markings (adopted from [18]).

The road markings can be extracted using image-based sensors like monocular or stereo vision cameras, or using LIDAR sensors. Among these, using monocular cameras is the cost-effective approach, although they lack the depth information. Stereo vision cameras can, however, provide the capability to infer the depth information and hence the ability to reconstruct three-dimensional scenarios for improved functionality, such as collision detection [4]. LIDAR sensors exploit the fact that road markings are painted using retroreflective paints. These extracted markings can then be used to extract the lane markings. However, LIDAR sensors are, similar to stereo vision cameras, far more expensive than monocular cameras. As such, seeking a trade-off between performance, reliability, and cost is an important activity in the design process. Treating cost effectiveness as the primary objective, we assume that the lane detection is performed on images obtained from a monocular camera system.

The literature on lane detection and tracking is considerably rich with a variety of techniques, covering various applications domains, including LDW, LKA, LCM, and CD. Some of these perform lane marking detection (for example, [19,20]) and track them while the rest perform only the detection (for example, [5,13,21]). In particular, we focus on techniques that solely rely on images or videos obtained from monocular vision cameras for lane marking detection followed by tracking. For instance, vision-based lane detection has been used for LDW in [5,11,12,14,22]. These approaches predominantly rely on information such as color, color cues, and edge-specific details. Color cues exploit the color contrast information between the lane markings and roads. However, the conditions have to be favorable for the differences in contrast to be realized by the lane marking algorithms. Conditions such as illumination, back lights, shadows, night lights, and weather condi-

tions, such as rain and snow, can significantly affect the performance of color-cue-based algorithms. One approach to overcome these limitations is to use the Hough transform along with color cues [23]. However, Hough transform works well when the potential candidate lines are straight and visible enough. Although some preprocessing can improve the detection [21], consistently differentiating lane boundaries from other artifacts, such as shadows and vehicles, is a challenge.

Inverse perspective Mapping (IPM) is another approach to determine the lane boundaries in LDW systems. The central idea behind IPM is to remove the perspective distortion of lines that are parallel in real world [11,18,22]. In order to do this, images are transformed from camera view to bird's eye view using camera parameters. During the transformation, the aspect ratios are retained so that gap or widths between lane boundaries are transformed appropriately. As such, the lane boundaries are still detectable in the transformed space. However, there are several downsides to this approach. Primarily, IPM is often used with fixed camera calibration parameters, and this may not always be optimal, owing to the surface conditions [24]. Furthermore, these transformations are computationally intensive [25], and as such, the real-time utility of these approaches needs careful implementation. Although these issues can reasonably be overcome by resorting to various techniques, such as calibration and adequate compute power systems [24–26], the main limitation is that the transformation is sensitive to obstacles on the road, such as vehicles, and to terrain conditions [27].

As lane markings are a pair of parallel lines, each pair should pass through a vanishing point [28]. This property can be exploited to eliminate and filter out the line segments that do not constitute lanes [29–31]. A number of approaches have been proposed in the literature for tracking a single lane, such as [14,17,31–35]. In [17], color, gradient, and line clustering information are used to improve the extraction of lane markings. In [36], an approach for lane boundary detection based on random finite sets and PHD filter is proposed as a multitarget tracking problem. In [32], a multilevel image processing and tracking framework is proposed for a monocular camera-based system. As such, it heavily relies on preprocessing of frames. Our approach also uses splines, but our tracking approach is significantly different to the one in [32]. In [33,34], techniques for personalized lane-change maneuvering are discussed. They use driver-specific behaviors, collected as part of the system, to improve the results. Although this can improve the results, such approaches are practically difficult to implement. In [35], the lane tracking is simplified by forming a midline of a single lane using B-splines. Although this approach may be useful over a short distance, conditions such as diverging lanes or missing lane markings will render the approach susceptible to bad approximations of midlines. This can easily lead to suboptimal results.

With recent advances in machine learning, particularly with supervised learning techniques such as deep learning, it is possible to engineer machine learning models to recognize lane markings. This possibility has been demonstrated in the literature [37–41]. In [38], a special convolutional neural network (CNN), termed spatial CNN (SCNN), was constructed for extracting the spatial correlation between objects in an image with the view of using that to establish the relative positioning of lane markings. In [39], the LaneNet consisting of two deep neural networks was constructed for lane detection. One of the networks detects lane marking edges, whereas the other network groups and clusters lane markings. The lane extraction work described in [38] relies on several integrated techniques, such as the YOLO framework [42] for object detection and convolutional patch networks (CPN) [43,44] for detecting road surfaces and regions of interest. Although these supervised techniques can offer a good result, the approach suffers from a number of issues. First, supervised learning techniques rely on labeled datasets or ground truth information. Although this appear to be trivial, these labels have to be made for each and every pixel that are to be classified as lane marking. Second, the real success of deep learning is based on the volume of data upon which the model is trained. Although the process of securing several thousands of images with labeled pixels can be automated, it is a time-consuming

process. Third, training requires substantial amount of compute time. Fourth, although various supervised learning techniques can offer good accuracy rates, the explainability of the machine learning models is still an upcoming area of research, and unlike general algorithms, deep neural networks lack the rigor of explainability. This is a serious concern where lives could be at risk. Finally, the accuracy rates are never sustained across different datasets. As such, the training process is a continuous one.

This paper aims to develop a tracking technique for a multilane tracking problem based on images/videos captured from a single camera mounted in front of a vehicle. The algorithm is designed to be real-time, robust, and cost-efficient in terms of sensors. To this end, we first model each lane marking as a spline, defined by a finite set of control points. By treating these splines (and thus the control points) as targets whose motions are defined during frame transitions, we develop a multitarget tracking problem with an appropriate motion model. The multilane tracking technique proposed in this paper is a precise amalgamation of several existing real-time and robust ideas in the pipeline with the addition of certain new ideas mentioned in the contribution.

We utilize the probabilistic Hough transformation [45] to perform an initial extraction of lane markings. This is then followed by a series of algorithms prior to treating the extracted lanes as targets. The first algorithm in the pipeline performs an initial grouping of extracted line segments into different splines. This is then followed by an algorithm, which encapsulates a number of subalgorithms, to differentiate the clutter from lane boundaries in a robust manner and to manage the evolution of trajectories of splines being tracked. We then devise a multitarget tracking algorithm based on a motion model that assumes that the transitions of splines across frames are at a constant rate. The overall solution can be considered as a carefully engineered pipeline of algorithms. In doing this, we make the following key contributions:

1. We develop an algorithm, based on a maximum a posteriori (MAP) estimator [46], to group and cluster different lane segments into unknown spline groups;
2. find intensity likelihood ratio of line segments and augment this ratio as a feature in a clustering and probabilistic data association (PDA) filter [47] to distinguish lane markings from clutter; and
3. propose a new, real-time, multiple extended target tracking (targets that shape and position changed simultaneously) algorithm that works with clutter existence based on the PDA filter to distinguish and track multiple spline shape lane-lines.

The remainder of this paper is organized as follows: In Section 2, we formulate the overall problem, and discuss our approach for solving each of the subproblems. This is then followed by Section 3, in which we discuss a set of preprocessing steps on the input images prior to using our framework of methods. The aspect of clustering and estimating control points to describe the splines, and two of our key algorithms for this purpose are discussed in Section 4. We then describe the techniques to track multiple splines using the IPDA filter in Section 5. The results of our evaluations are then presented in Section 6, and we discuss conclusions in Section 7.

2. Problem Formulation and Our Approach

2.1. Problem Formulation

To facilitate the process of deriving an overall approach and suitable algorithms, we use i as the index for the control points $i \in 0, 1, \ldots, N$, j as the lane index, and k as the frame index. For instance, the parameter $x_{i,j,k}$ denotes the ith control point for the jth lane on the kth frame. The notations used in this manuscript are given in Table 1.

With these notations in place, the overall problem of lane identification across a sequence of frames can be reformulated as follows:

1. Identification of Control Points: For a set of extracted lane markings on frame k, identify a set of control points that would uniquely describe each of the lane markings (assuming that lane markings do not cross each other);

2. Trajectory Management: Each candidate control point belongs to one of the lane markings, and over a period of time (across frames), these control points form distinctive trajectories if they are managed well. In order to manage these trajectories, which is a prerequisite for multitarget tracking, it is crucial to associate the control points to the trajectories they belong to; and
3. Multitarget Tracking: By using control points in each of the trajectories as pseudo measurements, formulating a multitarget tracking algorithm is essential for further extraction and identification of lane markings on the frames yet to be seen.

Table 1. Symbols adopted in this manuscript.

Symbol	Description
$x_{i,k,j}$	Single control point in lane-line(spline) j on frame k
$Z_{k,j}$	Pseudomeasurements (control points) for a spline j on frame k
R	Covariance of the measurement noise
Q	Covariance of the process noise
τ	Track index
F	State transition matrix
H	Measurement matrix
P_G	Gating probability
$e_{k,j}$	Event on the jth spline at the kth frame
$\Psi_{k,j}$	Line segment measurements for lane-line j

2.2. Our Approach

In addressing the overall problem outlined in Section 2.1, we decompose that into a number of subproblems, each of which handles a specific aspect of the overall lane detection problem across frames. The overall agenda is to form an automated processing pipeline, where each stage of the pipeline is underpinned by one or more algorithms. This processing pipeline is shown in Figure 2.

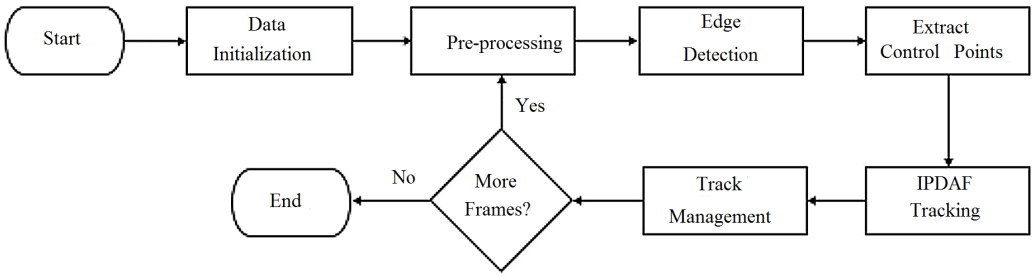


Figure 2. The flowchart of the proposed approach.

Each of these stages is discussed in the following sections.

3. Preprocessing

The key aspects of the preprocessing stage include edge detection, probabilistic Hough transform, and extraction of region of interest. We also use noise filtering before each of these stages to minimize the impact of noise amplification in the process.

3.1. Edge Detection

The basic edge detection in images is based on the convolution of a predetermined kernel with an image [48]. In our case, each frame forms an image. However, this basic approach for edge detection, which is a gradient finding exercise, picks up the gradients of the noise along with the lane markings. Although basic noise filtering, such as averaging or median filtering, can minimize these effects, they do not guard the edge detection from these artifacts. For this reason, we used the Canny edge detection [48], which incorporates Gaussian filtering as a precursor step to gradient calculation. More specifically, we used two 3×3 kernels, namely a Gaussian kernel H and an edge detection kernel K. For each

input frame $\mathcal{F}_{in}$, we calculated the output frame $\mathcal{F}_{out}$ as $\mathcal{F}_{out} = K * (H * \mathcal{F}'_{in})$, where $\mathcal{F}'_{in}$ denotes the noise-filtered version of $\mathcal{F}_{in}$, and $*$ operator denotes the convolution operation. We have also prefixed the values of the H (by fixing the variance).

3.2. Probabilistic Hough Transform

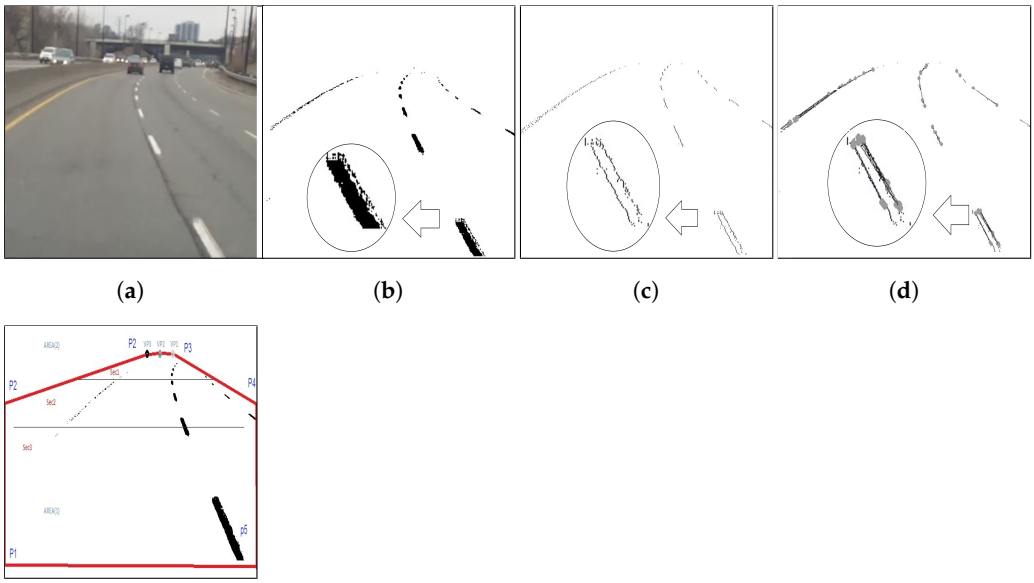
Although edge detection process brings out the edges in each frame, they do not have to correspond to straight lines in the real image, particularly the lanes in the partitioned tiles. In other words, among many pixels forming different edges, the key interest is on pixels that make up straight lines—lanes. An easier implementation of this is due to [49], where an accumulator matrix captures the intersections of various straight lines in an image. This matrix is then exhaustively searched for a maximum (and other decreasing maxima) to find straight lines. As such, the original Hough transformation process is computationally intensive. In our case, we adopted the probabilistic version of Hough transform [50], where only a subset of the edge points are selected through random sampling process, particularly when updating the accumulator matrix. With this approach, we reduce the amount of computation without considering all possible measurements.

3.3. Extraction of Regions of Interest

Although the probabilistic Hough transform can filter out unnecessary edges and lead to straight lines, the extracted straight lines do not have to represent only the lanes. In fact, the extracted straight lines can be anything, including lanes, edges of the vehicles, lampposts, and buildings. An easier approach to filter out irrelevant components is to use the vanishing points. Each pair of lane, unlike other straight lines in a frame, should have a vanishing point.

Vanishing points can be extracted by embedding an additional process after the probabilistic Hough transform process outlined above. Sinusoids that pass through all of the maxima points in Hough space should correspond to the vanishing point in the image plane. In particular, we extract vanishing points for each partition of the image. We then use these vanishing points to eliminate irrelevant straight line segments in the image and to form regions of interest. In addition to this, the area outside the vanishing line has no information that can aid lane boundary tracking, and can be removed.

In Figure 3, we show the outputs of different stages of the preprocessing.



(**a**) (**b**) (**c**) (**d**)

(**e**)

Figure 3. An example of preprocessing of image frames. (**a**) Raw image of the frame; (**b**) after selective segmentation; (**c**) after noise filtering and thinning; (**d**) after probabilistic Hough transform; (**e**) after extracting regions of interest.

4. Clustering and Identification of Control Points

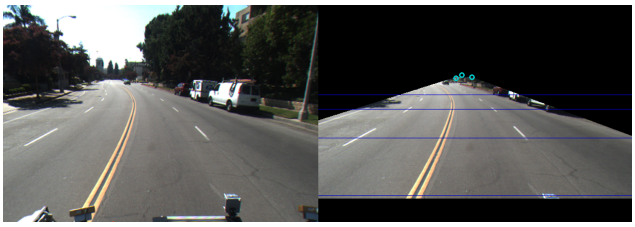
Identification of lane-line (spline) control points starts with:

1. Partitioning frame, finding line segments and likelihood ration.
2. Predict control points and validated measurements.
3. Update final control points using MAP estimator.

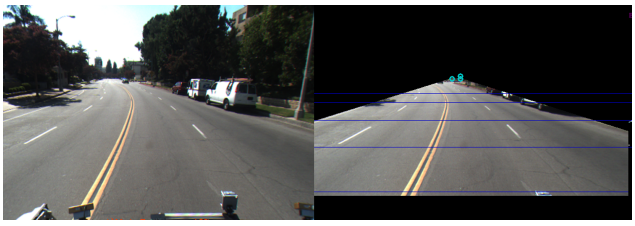
4.1. Frame Partitioning

Once the preprocessing is over, the next stage of the pipeline extracts the control points. Although we intend to identify a set of control points to model the lanes as splines, the process is much simpler if the splines are small in size and straight in shape. However, the extracted lane markings are seldom straight. One approach to address this issue is to partition each frame into n horizontal tiles, each with an experimentally determined height, so that lanes on each partition are near straight. Figure 4 shows the same image partitioned in two different ways: for two different values of n (namely $n = 3$ and $n = 4$), and with different partition heights.

However, considering the perspective characteristics of the camera and the distance of lanes from the camera, it is beneficial to have the heights of the partitions in increasing order toward the bottom of the frame. We experimentally determined that the extracted information is maximized for $n = 3$, such that $h_1 = \frac{1}{7}H$, $h_2 = \frac{2}{7}H$, and $h_3 = \frac{4}{7}H$, where H is the overall height of the region of interest (ROI). We use this configuration with the values of n and h_i $(i = 1, 2, 3)$ throughout the study conducted in this paper.



(**a**) Partitioning for $n = 3$.



(**b**) Partitioning for $n = 4$.

Figure 4. Two different examples of partitioning outputs (right) for the same input image (left). (**a**) $n = 3$ and $h_1 = \frac{1}{7}H$, $h_2 = \frac{2}{7}H$ and $h_3 = \frac{4}{7}H$; (**b**) $n = 4$ and $h_1 = \frac{1}{11}H$, $h_2 = \frac{2}{11}H$, $h_3 = \frac{3}{11}H$, and $h_4 = \frac{5}{11}H$.

4.2. Intensity Likelihood Ratio of a Line Segment

For each of the partitions, we apply the probabilistic Hough transform to extract the lane markings. However, the extraction process, akin to edge in most of the detection techniques, produces a number of broken, small, noncontinuous, and irrelevant line segments. As such, one of the key challenges following the extraction process is to distinguish the lane markings from background noise and clutter. To render a more robust high-fidelity approach toward clutter and noise management, we augment the extractions with underlying intensity values. More specifically, we define the number of edge points that lie in an extended line segment s $(s = 1, \ldots, n)$ as the intensity. The intensity can be extended to cover a set of line segments or a number of pseudomeasurements belonging to a curve. The intensity of an extended line segment is represented as a likelihood ratio, which we define below.

Let $p_0(f_j)$ be the probability density function (PDF) of the noise only, and $p_1(f_j)$ be the target originated line-segment detections before thresholding. Furthermore, let D_0 and D_1 be the scale parameters for false alarms and clutter, and target, respectively. These scale parameters are dependent on the minimum number of points used in the Hough transform. The noise only and target originated measurement density functions are

$$p_0(f_j) = \frac{f_j}{D_0^2} e^{\left(\frac{-f_j^2}{2D_0^2}\right)} \tag{1}$$

$$p_1(f_j) = \frac{f_j}{D_1^2} e^{\left(\frac{-f_j^2}{2D_1^2}\right)} \tag{2}$$

where $f_j \geq 0$ is the intensity of the candidate measurements j. Furthermore, let $\gamma = \gamma_{det}$ be the threshold to declare a detection. The probabilities of detection (P_D) and false alarm (P_{FA}) can be computed as follows:

$$\begin{aligned} P_D &= \int_{\gamma}^{\infty} p_1(f_j) df_j \\ &= e^{\frac{-\gamma^2}{2D_1^2}} \end{aligned} \tag{3}$$

$$\begin{aligned} P_{FA} &= \int_{\gamma}^{\infty} p_0(f_j) df_j \\ &= e^{\frac{-\gamma^2}{2D_0^2}} \end{aligned} \tag{4}$$

Although the probability of detection, P_D, can be increased by lowering γ, it will increases P_{FA}. Hence, the choice of γ cannot be arbitrary. With these, the corresponding probability density functions after thresholding become

$$\begin{aligned} p_0^{\gamma}(f_j) &= \frac{1}{P_{FA}} p_0(f_j) \\ &= \frac{f_j}{P_{FA} D_0^2} e^{\left(\frac{-f_j^2}{2D_0^2}\right)} \end{aligned} \tag{5}$$

$$\begin{aligned} p_1^{\gamma}(f_j) &= \frac{1}{P_D} p_1(f_j) \\ &= \frac{f_j}{P_D D_1^2} e^{\left(\frac{-f_j^2}{2D_1^2}\right)} \end{aligned} \tag{6}$$

where $p_0^{\gamma}(f_j)$ and $p_1^{\gamma}(f_j)$ are the probability density functions of the validated measurement ψ_j (for $j = 1, \ldots, m$) that are due to noise only and originating from the target, respectively.

Considering Equations (5) and (6), the line segment intensity likelihood ratio e_j, which is the likelihood ratio of measurement ψ_j with intensity of f_j edge pixels originating from target rather than clutter, can be defined as

$$\begin{aligned} e_j(k) &= \frac{p_1^{\gamma}(f_j)}{p_0^{\gamma}(f_j)} \\ &= \frac{P_{FA} D_0^2}{P_D D_1^2} e^{f_j^2 \left(\frac{D_1^2 - D_0^2}{2D_0^2 D_1^2}\right)} \end{aligned} \tag{7}$$

4.3. Pseudomeasurements

Pseudomeasurements $Z^{\tau}_{k,j}$ is a set of the control points for track τ and lane j in frame k. More specifically,

$$Z^{\tau}_{k,j} \quad = \quad [x^{\tau}_{1,j,k}, x^{\tau}_{2,j,k}, x^{\tau}_{3,j,k}, x^{\tau}_{4,j,k}] \tag{8}$$

Furthermore, let $\psi^k_s(j)$ denote the extended line segment in section s, at time step k, for lane j—that is, $\psi^k_s(j)$ abstracts away a number of pseudomeasurements for each (extended) line segment. Each such measurement is a two-element vector, with one capturing the pseudomeasurement $Z^{\tau}_{k,j}$ and the other one representing the intensity of the extended line segment as a likelihood ratio, $e_j(k)$.

4.4. MAP Estimator for Measurements with Intensity Feature

Although we expect the pseudomeasurements to almost model the lane-lines, in reality, a number of factors make this process as challenging. Examples include, but are not limited to, missed detection, nondeterministic nature of the preprocessing, and noisy measurements due to clutter. Therefore, it is essential to model these imperfections as part of the process.

To simplify the analysis and derivation, we assume that measurements that originate from targets at a particular sampling instant are received by the sensor only once with probability of detection P_D. The measurement equation can be written as follows:

$$\psi(j) = x + w(j) \tag{9}$$

where $j = 1, \ldots, m$, $w(j)$ is the measurement noise, and $\mathbf{x} = [x_1, x_2]'$ is the unknown value that we are aiming to estimate in the presence of the measurement noise.

We also assume that the measurement noise is independent and zero mean Gaussian distributed with covariance R. In our case, various preprocessing stages, such as thinning and Hough transform, contribute towards R. Thus, $w(j) \sim \mathcal{N}(0, R)$, where

$$R = \begin{bmatrix} \sigma^2_{11} & 0 \\ 0 & \sigma^2_{12} \end{bmatrix} \tag{10}$$

Because of the condition of the road and perspective effect of the camera lens for values of σ^2_{11} and σ^2_{12}, we would expect more deviation in the bottom part that is closer to the camera compared to the top. We also assume the measurements ψ to be normally distributed around x with covariance R and the prior probabilities $p(x)$ to be normally distributed around the predicted measurement $\bar{x}$ with a covariance Q. Thus, $\psi \sim \mathcal{N}(x, R)$ and $p(x) \sim \mathcal{N}(\bar{x}, Q)$, where

$$Q = \begin{bmatrix} \sigma^2_{01} & 0 \\ 0 & \sigma^2_{02} \end{bmatrix} \tag{11}$$

Again, similar to R, the perspective effects of the camera influences the values of σ^2_{01} and σ^2_{02} to be skewed toward the bottom part of the frame. Furthermore, the covariance Q is often linked to the curvature κ of the road. Assuming the maximum standard curvature of highways as a constant parameter, the posterior measurement density would be

$$p(x\Psi) \stackrel{\Delta}{=} \frac{1}{c}(p(\Psi x)p(x)) \tag{12}$$

Since the measurement and prior noises are assumed to be Gaussian, for a single measurement, (i.e., $m = 1$), $\psi(1) = \psi_1$ can be expressed as:

$$\begin{aligned} p(x\psi_1) &\triangleq \frac{1}{c}(p(\psi 1 x)p(x)) \\ &= \frac{1}{c}\mathcal{N}(\psi 1; x, R)\mathcal{N}(x; \bar{x}, Q) \\ &= \frac{1}{c}\mathcal{N}(x; \xi(\psi_1), \mathcal{R}) \end{aligned} \tag{13}$$

where

$$\xi(\psi_1) = \frac{Q}{R+Q}\bar{x} + \frac{R}{R+Q}\psi_1$$

and

$$\mathcal{R} = \frac{RQ}{R+Q}$$

For a Gaussian distribution, the mean is the optimal maximization value $\hat{x}$. Hence,

$$\hat{x} = \bar{x} + \frac{R}{R+Q}(\psi_1 - \bar{x}) \tag{14}$$

For $m > 1$, the optimal maximized value can be derived using the total probability and combined residuals as follows:

$$\hat{x} = \bar{x} + \frac{R}{R+Q}\sum_{j=1}^{m}\beta_j(\psi(j) - \bar{x}) \tag{15}$$

where β_j is association probability, which we define as (see Appendix A for derivations). where P_D and P_G are probabilities of detection and gating, respectively, m_k is the number of validated detections at time k, e_j is the intensity of the extended line segments as a likelihood ratio, and $\mathcal{L}_j$ is the probability density function for the correct measurement without the intensity feature, defined as

$$\mathcal{L}_j = \frac{1}{P_G}\mathcal{N}(\psi_j, \bar{x}, S)$$

where $S = R + Q$, and $\bar{x}$ is the prior information.

4.5. Clustering Algorithm for Finding Control Points

Ideally, each partition will have a sufficient number of full measurements $\psi^k(j)$ so that a spline can be fitted over those measurements. However, in reality, this is seldom the case. The associated challenges are dealt with here using an algorithm that estimates the control points based on the available set of measurements. In particular, we use the MAP estimator (MAPE) to find the optimal control points. These aspects are handled by two algorithms, Algorithms 1 and 2, which are outlined and discussed in detailed below.

Algorithm 1 handles each partition separately, but by extending the line segments into the next partition wherever needed. For a given partition s, it estimates the control points for each line, $\mathbf{x}_{i,s}$, using the curvature κ. Then, the overall set of lines L is used to estimate the control points for that partition using the MAP estimator (see Algorithm 2). These control points are accumulated into A as a list. Notice that the *Predict()* function finds predicted control points for each individual line segment l using curvature vector κ.

Algorithm 1 Control Points Estimator

▷ Input: κ, N, Ψ
▷ Output: A `//Set of control points`
▷ Section variables : P_i, R_i, S_i, ψ_i
▷ κ : Vector of curvature
▷ ψ_i : Sets of all extracted lines in partition i
▷ N : Number of partitions
▷ P_i : Prior noise covariance
▷ R_i : Measurement noise covariance
▷ S_i : Set of partitions indexes eliminating i
for i=1; i<N; i++ **do**
  for each $l \in \psi_i$ **do**
    for each $S_i \in \{1..N\} - \{i\}$ **do**
      `Initialize(`P_{S_i}, R_{S_i}`)`
      $\bar{x}_{S_i,l} \leftarrow$`Predict(`$\kappa, l$`)`
      $\hat{x}_{S_i,l} \leftarrow$`MAPE(`$\bar{x}_l, \psi_{S_i}, R_{S_i}, P_{S_i}$`)` `//Update`
    end for
    $A \leftarrow \begin{bmatrix} A & \hat{x}_l \end{bmatrix}$
  end for
  `A=RemoveSimilarCurves(`A`)`
end for

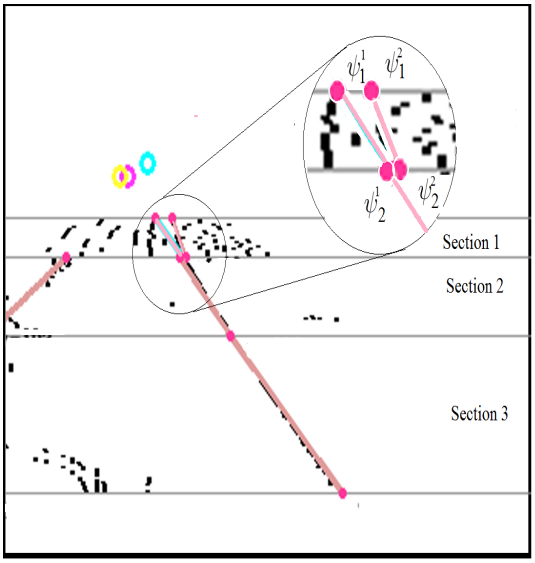
Algorithm 2 MAPE.

▷ Inputs: $\bar{x}, \psi, R, P$
▷ Output: $\hat{x}$
▷ Section variables : P, R
▷ $\bar{x}$: Priors
▷ ψ : Measurements
▷ P : Prior noise covariance
▷ R : Measurement noise covariance
$\psi_{validated} \leftarrow$ `Validated-Measurements(`$\bar{x}, \psi, R, P$`)`
$m \leftarrow \psi_{validated}$
for j=0; j<m; j++ **do**
  $r_j \leftarrow \psi(j) - \bar{\mathbf{x}}$
  $\beta_0 \leftarrow \frac{(1-P_D P_G)}{C}$
  $\beta_j \leftarrow \frac{\mathcal{L}_j e_j}{C}$
end for
$\hat{x} = \bar{x} + \frac{R}{R+Q} \sum_{j=1}^{m} \beta_j r_j$

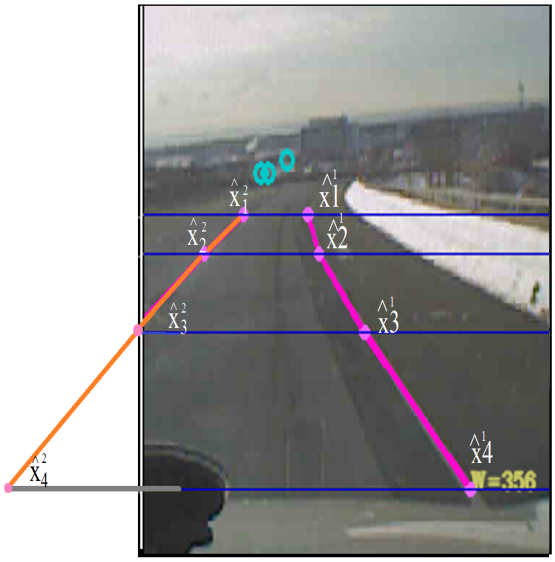
Algorithm 2 combines both the data association and the posteriori PDF to adjust the estimated control points. In particular, it uses the IPDA-specific target-to-track association probabilities (covering both the track existence and non-existence), β_0 and β_j for finding the control points based on candidate control points $\bar{x}$ and measurements Ψ. More specifically,

$$\begin{aligned} \hat{x} &= \underset{x}{\text{argmax}}\ p(x\mathbf{\Psi}) \\ &= \underset{x}{\text{argmax}}[p(\mathbf{\Psi}x)p(x)] \end{aligned} \tag{16}$$

Validated-Measurements() function uses normalized distance to validate the line segments belonging to each spline. We show a sample outcome of these algorithms in Figure 5. We first show two endpoint measurements (ψ_1^1, ψ_2^1) and (ψ_1^2, ψ_2^2) (Figure 5a). These points are then corrected using the above algorithms to output corrected control points $\hat{x_1^1}$ and $\hat{x_2^1}$ (Figure 5b).



(**a**)



(**b**)

Figure 5. An example of control point estimation/correction and extended line segments. (**a**) Control point estimation/correction based on measurements; (**b**) grouped line segments based on the measurements.

5. Multilane Tracking Using IPDA

PDA filter is similar to Kaman filter in the dynamic model and prediction, but in the update step, it uses the sum of weighted measurements to deal with clutter. We used integrated PDA (IPDA) [51] filter in the multi-target platform in a new way to combine initialization and track management for tracking multiple unknown numbers of extended targets (splines) with augmented intensity as a new feature inside association likelihood to deal with clutters, Figure 6.

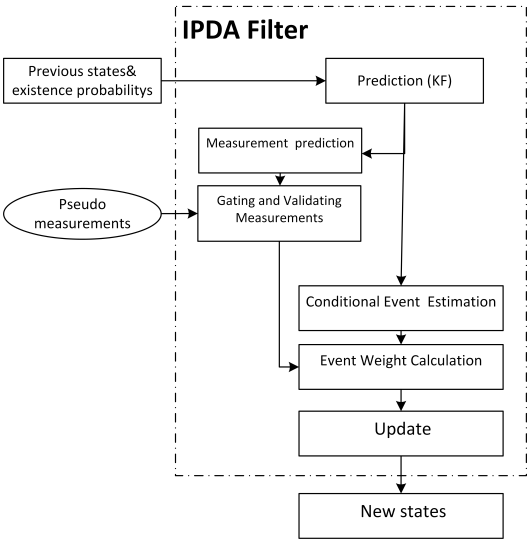


Figure 6. IPDAF steps.

5.1. Preliminaries

As stated above, we assume that control points are moving at constant velocity, and thus our dynamic model is a constant velocity model. With this, our state vector for track τ in frame k becomes

$$x_k^\tau = \begin{bmatrix} \mathbf{x}_k^\tau \\ \dot{\mathbf{x}}_k^\tau \end{bmatrix} \tag{17}$$

where

$$\mathbf{x}_k^\tau = \begin{bmatrix} x_{1,k}^\tau \\ x_{2,k}^\tau \\ x_{3,k}^\tau \\ x_{4,k}^\tau \end{bmatrix} \text{ and } \dot{\mathbf{x}}_k^\tau = \begin{bmatrix} \dot{x}_{1,k}^\tau \\ \dot{x}_{2,k}^\tau \\ \dot{x}_{3,k}^\tau \\ \dot{x}_{4,k}^\tau \end{bmatrix}$$

and the intensity feature f augmented (pseudo)measurement vector $Z_{k,j}^{\tau,f}$ is

$$\begin{aligned} Z_k^{\tau,f} &= \begin{bmatrix} x_{1,k}^\tau \\ x_{2,k}^\tau \\ x_{3,k}^\tau \\ x_{4,k}^\tau \\ f_k^\tau \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{x}_k^\tau \\ \mathbf{f}_k^\tau \end{bmatrix} \end{aligned} \tag{18}$$

With these, the state evolution and measurement updates for frame (time) index k become

$$x_k^\tau = Fx_{k-1}^\tau + Gv_{k-1}^\tau + \mathcal{G}u_{k-1}^\tau \tag{19}$$

and

$$Z_k^{\tau,f} = Hx_k^\tau + \omega_k^\tau \tag{20}$$

where F, G, and H are state transition, control input, and and observation matrices, respectively, and ν and ω are measurement and process noises, respectively. In our case, we preset F and G to

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{21}$$

$$G = \begin{bmatrix} \frac{\Delta t^2}{2} & 0 & 0 & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 & 0 \\ 0 & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 0 & 0 & \frac{\Delta t^2}{2} \\ \Delta t & 0 & 0 & 0 \\ 0 & \Delta t & 0 & 0 \\ 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & \Delta t \end{bmatrix} \tag{22}$$

To minimize the variability of the outcomes of the lane extraction process, or its dependencies on the highly variable surrounding contexts, the selection process of the control points has to be adaptive enough to account the context information. For instance, when comparing the lane extraction process on roads with varying terrain conditions opposed to highways, it is desirable if the process noise is more amenable to high variance on the spatial distribution of control points. One approach to bring the adaptiveness into this process is to introduce adaptive process and measurement noises. Furthermore, due to partitioning of frames and due to the perspective effects of the camera, control points on the bottom most partition are closer to the camera than the control points on the top-most partition of a frame. As such, despite the constant velocity model, the rates of the spatial variations of control points are different. Considering different noise variance associated with the acceleration of control points will address this issue. As such, the diagonal form of the process noise ν_k can be expressed as

$$q = \begin{bmatrix} \ddot{x}_{1,k}^{\tau} & 0 & 0 & 0 \\ 0 & \ddot{x}_{2,k}^{\tau} & 0 & 0 \\ 0 & 0 & \ddot{x}_{3,k}^{\tau} & 0 \\ 0 & 0 & 0 & \ddot{x}_{4,k}^{\tau} \end{bmatrix} \tag{23}$$

where $\ddot{x}_{i,k}^{\tau}$ for $(i = 1, 2, 3, 4)$ are the maximum acceleration of each control point. The process noise covariance Q can be expressed as

$$Q = GqG^T \tag{24}$$

where

$$diag_0(Q) = \begin{bmatrix} \ddot{x}_1 \frac{\Delta t^4}{4} \\ \ddot{x}_2 \frac{\Delta t^4}{4} \\ \ddot{x}_3 \frac{\Delta t^4}{4} \\ \ddot{x}_4 \frac{\Delta t^4}{4} \\ \ddot{x}_1 \Delta t^2 \\ \ddot{x}_2 \Delta t^2 \\ \ddot{x}_3 \Delta t^2 \\ \ddot{x}_4 \Delta t^2 \end{bmatrix}^T \tag{25}$$

$$diag_{-4}(Q) = \begin{bmatrix} \ddot{x}_1 \frac{\Delta t^3}{2} \\ \ddot{x}_2 \frac{\Delta t^3}{2} \\ \ddot{x}_3 \frac{\Delta t^3}{2} \\ \ddot{x}_4 \frac{\Delta t^3}{2} \end{bmatrix}^T \tag{26}$$

and

$$diag_{+4}(Q) = \begin{bmatrix} \ddot{x}_1 \frac{\Delta t^3}{2} \\ \ddot{x}_2 \frac{\Delta t^3}{2} \\ \ddot{x}_3 \frac{\Delta t^3}{2} \\ \ddot{x}_4 \frac{\Delta t^3}{2} \end{bmatrix}^T \tag{27}$$

where $diag_i(Q)$ is the ith subdiagonal of Q, with $i = 0$ denoting the main diagonal, $-i$ and $+i$ denoting ith subdiagonals below and above the main diagonal, respectively. Similarly, the measurement noise covariance R is

$$R = \begin{bmatrix} \sigma_{1x}^2 & 0 & 0 & 0 \\ 0 & \sigma_{2x}^2 & 0 & 0 \\ 0 & 0 & \sigma_{3x}^2 & 0 \\ 0 & 0 & 0 & \sigma_{4x}^2 \end{bmatrix} \tag{28}$$

5.2. Multilane Tracking Using IPDAF

The IPDA filter [51,52] offers an augmented information towards track maintenance apart from state estimation of tracks. Instead of assuming the existence of targets as a hard-wired probability, the IPDA filter offers a choice incorporating the track quality measure into the tracking process.

In the context of PDA algorithm, for a each validated measurement, the association probability (with each track) is calculated. To this end, the association probability $\beta_i^\tau(k)$ accounts for the probability of associating a measurement i to track τ_k, feature intensity of $f_i^\tau(k)$, and the likelihood ratio of associating a line with a feature measurement $e_i^\tau(k)$. These probabilistic information are used to associate new measurements to the targets. Given a linear dynamic model, and an IPDAF based on [47], the state and measurement equations becomes

$$\hat{x}(kk) = E[x(k)Z^k] = \sum_{i=0}^{m_k} \hat{x}_i(kk)\beta_i(k) \tag{29}$$

where $\hat{x}_i(kk)$ is the updated state conditioned on the event that the ith validated measurement being correct, and $\beta_i(k)$ is the probability of associating a measurement i with feature value of $f_i(k)$ to track k. The association probability for a set of m_k gated or validated measurements with features $f_i(k)$ can be expressed as

$$\beta_i^\tau(k) = P\{\epsilon_i(k)Z_k^\tau, f^\tau(k), m_k\} \tag{30}$$

where $\epsilon_i(k)$ is the event described in Appendix A. In our case, we assume that each detected lane boundary measurement i has a feature of intensity $f_i(k)$. With reference to (A13), the feature likelihood $e_i(k)$ can incorporated into the PDA algorithm as follows [53]:

$$\begin{aligned} \beta_i(k) &= P\{\epsilon_i(k)Z^k, m_k\} \\ &= \begin{cases} \frac{\mathcal{L}_i(k)e_i(k)}{1-P_D P_G+\sum_{i=1}^{m(k)} \mathcal{L}_j(k)e_i(k)}, \forall i \neq 0 \\ \frac{1-P_D P_G}{1-P_D P_G+\sum_{i=1}^{m(k)} \mathcal{L}_j(k)e_i(k)}, i = 0 \end{cases} \end{aligned} \tag{31}$$

where $i = \{0, 1, \ldots, m(k)\}$.

The overall IPDAF algorithm here embodies a traditional PDAF algorithm with special initialization and the termination steps. This is outlined in Algorithm A1 in the Appendix B. The underlying aspects of the multilane tracking follow the principles of multitarget tracking as in [47], and are discussed in the following subsections.

5.2.1. Track Initialization

The track initialization process (for each track) can rely on one or two seed points. In the one-point initialization method, position can be initialized from a single observation with zero velocity vector. Due to [54],

$$diag(P(00))^T = \begin{bmatrix} \sigma_{1x}^2 \\ \sigma_{2x}^2 \\ \sigma_{3x}^2 \\ \sigma_{4x}^2 \\ (\frac{V_{max}}{2})^2 \\ (\frac{V_{max}}{2})^2 \\ (\frac{V_{max}}{2})^2 \\ (\frac{V_{max}}{2})^2 \end{bmatrix} \tag{32}$$

and

$$\hat{x}(00)^T = \begin{bmatrix} x_{1_z} \\ x_{2_z} \\ x_{3_z} \\ x_{4_z} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{33}$$

This initialization allows the standard gating to be used during the following time step.

5.2.2. Measurement Prediction

For each track, the state vector, the measurements, and the state covariance matrices are predicted ahead as in the standard Kalman filtering. i.e.,

$$\hat{x}_{kk-1} = F_{k-1}\hat{x}_{k-1k-1} \tag{34}$$
$$P_{kk-1} = F_{k-1}P_{k-1k-1}F'_{k-1} + Q_{k-1} \tag{35}$$
$$\hat{z}_{kk-1} = H_k\hat{x}_{kk-1} \tag{36}$$
$$S_k = H_kP_{kk-1}H'_k + R_k \tag{37}$$

with the assumption of Gaussian posterior for $p(x)$ as

$$p[x_{k-1}Z^{k-1}] = \mathcal{N}(x_{k-1}; \hat{x}_{k-1k-1}, P_{k-1k-1}) \tag{38}$$

5.2.3. Measurement Gating

For each track, a validation gate is setup around the predicted measurement to select the candidate measurements for the data association. The size of the validation gate is correlated to the innovation covariance and the measurement noise. As per (38), at most, one of the validated measurements can be assigned to the target. The measurements outside the gate are assumed to be false alarms or measurements belonging to other targets. The validation region is the elliptical shape as follows:

$$V(k, \gamma) = \{z : [z - \hat{z}_{kk-1}]'S_k^{-1}[z - \hat{z}_{kk-1}] \leq \gamma\}, \tag{39}$$

where n_z is the dimension of measurement vector representing the degrees of freedom,

and γ is the gating threshold. Here, the gating threshold γ is a chi-squared distribution, parameterized by the probability of gating P_G, and by the degrees of freedom n_z [46].

5.2.4. Data Association

An incoming measurement at time index k, $z_i(k)$ with feature $f_i(k)$ is associated to track a τ, based on the association probability given by (31)

$$\beta_i(k) = \begin{cases} \frac{\mathcal{L}_i(k)e_i(k)}{1-P_DP_G+\sum_{i=1}^{m(k)}\mathcal{L}_j(k)e_i(k)}, \forall i \neq 0 \\ \frac{1-P_DP_G}{1-P_DP_G+\sum_{i=1}^{m(k)}\mathcal{L}_j(k)e_i(k)}, i = 0 \end{cases}$$

Here, there are two likelihood ratios of interest: $L_i(k)$ and $e_i(k)$. The former is the likelihood ratio of the incoming measurement $z_i(k)$. This is defined as

$$\mathcal{L}_i(k) = \frac{\mathcal{N}[z_i(k); \hat{z}(kk-1), S(k)]P_D}{\lambda} \tag{40}$$

where λ is the uniform density of the location of false measurements. The second parameter of interest, $e_i(k)$, is the likelihood ratio of measurement z_i^τ with feature f_i^τ of the track τ. This is defined as

$$e_i(k) = \frac{p_1^\tau(f_i)}{p_0^\tau(f_i)} \tag{41}$$

Both of these measurements, $z_i(k)$ and z_i^τ, are expected to originate from the target and not from the clutter.

5.2.5. State Update

The state, gain, and covariance update equations of the PDAF are

$$\hat{x}(kk) = \hat{x}(kk-1) + W(k)\mathcal{V}(k) \tag{42}$$

$$W(k) = P(kk-1)H(k)'S(k)^{-1} \tag{43}$$

$$P(kk) = \beta_0(k)P(kk-1) + (1-\beta_0(k))P^c(kk) + \tilde{P}(k) \tag{44}$$

where $W(k)$ is the Kalman gain, and $\mathcal{V}(k)$ is the combined innovation defined by

$$\mathcal{V}(k) = \sum_{i=1}^{m(k)} \beta_i(k)\mathcal{V}_i(k) \tag{45}$$

where m_k is the number of measurements inside the gating region, and $\beta_0(k)$ is the probability of all measurements being incorrect at time index k. With no information on which of the m_k measurements being correct or incorrect, the correct updated covariance can be expressed as

$$P^c(kk) = P(kk-1) - W(k)S(k)W(k)' \tag{46}$$

$$\tilde{P}(k) = W(k)\Big\{\sum_{i=1}^{m(k)} \beta_i(k)\nu_i(k)\nu_i(k)' \tag{47}$$

$$-\nu(k)\nu(k)'\Big\}W(k)' \tag{48}$$

5.2.6. Track Management

During the course of tracking, several tracks are maintained in parallel, and their states are continuously updated upon receiving measurements. A track can be in three different states: tentative, confirmed and terminated. During the initialization phase, every unassociated measurements will form a tentative track. However, upon following detections or measurements, and gating operations, the tracks will begin to form and

their status will be updated as confirmed. However, if no further measurement to track associations are possible, a possibility when no detections are observed from the target responsible for the track, corresponding track is terminated. For the case where $P_D < 1$, it is essential to check the quality of measurement to track association before engaging in track status update. One of the approaches for assessing the quality of measurement to track association is the goodness of fitting. The goodness of fitting, often represented by the log-likelihood function of the track, can be expressed as a recursive function as follows [46]:

$$\lambda(k) = \lambda(k-1) + \mathcal{V}(k)'S(k)^{-1}\mathcal{V}(k) \tag{49}$$

where $\mathcal{V}(k)$ is the innovation matrix, and $S(k)$ is the covariance of the innovation matrix $\mathcal{V}(k)$. The last term in (49) has a chi-squared density with n_z degrees of freedom, where n_z is the dimensionality of the measurement vector. As the innovations are independent, the log-likelihood function at time k is a chi-squared distributed with kn_z degrees of freedom. This is actually a measure of the goodness of fit to the assumed target model. Thus, the test function for keeping (or terminating) a track can be expressed as

$$\lambda^k \leq \lambda^k_{max} \tag{50}$$

$$\lambda^k_{max} = X^2_{kn_z}(1-\alpha) \tag{51}$$

where the tail probability α is the probability that a true track will be rejected. In our case, this threshold is around 0.01. However, the actual state transitions are performed through more rigorous checks. In our case, we maintain a number of (hidden) measurement and association counters for each track. These counters are used to assess the activeness of the measurement-track association.

6. Experiments and Evaluations

The proposed algorithm has been evaluated against two different baselines: model- and machine-learning-based implementations. The proposed algorithm was implemented using the OpenCV library in `C++`. The algorithms were tested on a system with Intel i7 CPU, clocked at 2.9 GHz with 16 GB RAM.

6.1. Evaluation against Model-Based Approaches

The proposed approach was benchmarked against two model-based methods, namely [18,55] using the Caltech Lane dataset [18]. The dataset has four video clips taken at different times of a day around the urban area of Pasadena in California. Each of these video clips has a resolution of 640×480, and covers various lighting and illumination conditions, writings along with lane markings (Clip#1), sun glint and different pavement types (Clip#2), shadows and crosswalks (Clip#3), and congested settings (Clip#4). As such, they are reasonably representative enough of various challenging conditions for tracking lane markings. In total, 1224 frames and 4399 lane boundaries were processed. The details of these video clips are given in Table 2. Notice that ROI parameters needs to be set in our method similar to [18,55]. Our algorithm does not need camera parameters, but since [18,55] use IPM mapping, they need those parameters to be set.

During evaluation, we computed the true and false positive rates (TPR and FPR, respectively), where the TPR is the ratio of the number of detected lane boundaries to the number of target lane boundaries and the FPR is the ratio of the number of false positives to the number of target lane boundaries. The frames were processed at the rate of seven frames per second similar to that of other methods in the literature [18,55]. In addition to TPR and FPR metrics, we also included another metric, false positives per frame (FP/frame or FPF), which is an average of false positives across all frames. One could equally consider the true positives per frame rate as well. We used the TPR, FPR and FP/Frame as the metrics of evaluation on for model-based approaches. The results of the evaluation are shown in Table 3.

Noting that higher TPR, lower FPR, and lower FP/frame values are desirable, we highlight the best (boldface) and second best results (underlined) in the results outlined in Table 3. A number of observations can be made here:

- The TPR performance of the proposed algorithm is consistently higher than those of the other two algorithms throughout all video clips. The performance of the proposed algorithm over Clip#3 is significantly higher than that of the method in [55];
- The FPR performance of the proposed algorithm is always better than that of the method in [18];
- The FPR performance of the proposed algorithm performs better than that of the method in [55] except for Clip#4. One potential reason for this sub-optimal performance for this clip can be attributed to the difficulties in association in congested settings; and
- The FP/Frame performance is mixed across the cases.

In overall, the proposed approach outperforms the other two methods across all cases. The overall performance differences are 5%, 2%, and 2% for TPR, FPR, and FPF cases when compared against the second best version, namely the method in [55]. To ensure the operation, we manually analyzed the datasets to label all visible and invisible lane markings as target lane markings.

Table 2. Caltech dataset used in the evaluation.

Clip ID	Clip Name	No. of Frames	No. of Lane Boundaries
1	cordoval1	250	975
2	cordoval2	406	1131
3	washington1	336	1329
4	washington2	232	964

Table 3. Comparison of our approach with other lane detection algorithms.

	Method in [18]			**Method in [55]**			**Proposed**		
	TPR	**FPR**	**FP/Frame**	**TPR**	**FPR**	**FPF/Frame**	**TPR**	**FPR**	**FP/Frame**
Clip#1	0.823	0.099	0.384	0.892	0.125	0.488	0.899	0.093	0.405
Clip#2	0.839	0.224	0.672	0.865	0.209	0.628	0.870	0.166	0.535
Clip#3	0.934	0.148	0.542	0.850	0.111	0.408	0.937	0.107	0.455
Clip#4	0.890	0.102	0.418	0.898	0.063	0.259	0.974	0.099	0.424
Overall	0.871	0.148	0.529	<u>0.874</u>	<u>0.131</u>	<u>0.469</u>	**0.920**	**0.116**	**0.454**

6.2. Evaluation against Deep Learning-Based Approaches

In this setting, we evaluated the proposed method against the spatially convolutional neural network (SCNN) method [38] using the TuSimple data-set [56]. The TuSimple dataset has about 7000 one-second-long video clips, each with 20 frames. The ground-truth information is available only for the last frame (frame 20) of each clip, including height and width values corresponding to lanes. Although the TuSimple dataset includes a number of road conditions, such as, straight lanes, curvy lanes, splitting and merging lanes, and shadows, we used only the straight and curvy lane conditions. Notice that like all fully supervised models the algorithm in [38] uses the trained parameters coming from training process, but we only use parameters for ROI in preprocessing step.

In addition to using TP and FN, we also use the accuracy and inference time as additional metrics of our evaluation. The overall results are shown in Table 4.

From these results, it can be seen that the proposed method outperforms the baseline method across all metrics. More specifically, the proposed method offers additional 4% improvement in accuracy along with ninefold speedup.

Table 4. Proposed approach compared with one of the deep-learning-based models (SCNN).

	SCNN [38]				Proposed			
	FPR	**FNR**	**Accuracy**	**Frame/s**	**FPR**	**FNR**	**Accuracy**	**Frame/s**
Straight Highway#1	0.166	0.250	0.855	0.71	0.160	0.250	0.880	6.5
Curvy Highway#2	0.479	0.416	0.825	0.72	0.166	0.250	0.861	6.5
Overall	0.375	0.361	0.836	0.71	**0.160**	**0.250**	**0.869**	**6.5**

7. Conclusions

In this paper, we proposed a novel approach for multilane detection. By using the intensity feature in conjunction with the probabilistic Hough transform, we first formulated an algorithm for detecting and correctly grouping multiple lane markings. By using these lane marking as splines, we then identify a set of control points, which then get tracked over time across frames. Our evaluations, covering both model-based and machine-learning-based approaches show that it can easily outperform the model-based approaches while being suboptimal compared to the deep-learning-based approaches. However, there are a number of issues remain to be addressed. For instance, machine learning models do not provide any explanation or reasons for their decisions compared to filter-based approaches like one presented here. As such, the proposed approach embodies sufficient explainability for its actions. Further investigations are needed to establish how the performance of the proposed approach can be improved.

Author Contributions: Conceptualization, B.A.; Data curation, B.A.; Formal analysis, B.A.; Funding acquisition, R.L. and K.T.; Investigation, K.T.; Project administration, B.A. and K.T.; Resources, B.A.; Software, B.A.; Supervision, K.T.; Validation, B.A.; Visualization, B.A.; Writing–original draft, B.A.; Writing–review & editing, J.T. and K.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Association Probability for Lane Measurements

Assuming there are m detections at time k, when finding the association probability β_j between a measurement and a target (lane j), the overall event $\epsilon(j)$ is comprised of two mutually exclusive events: either $\epsilon(j)$ is such that the measurement $\psi(j)$ is from the target, for $j = 1, \ldots, m$, or all measurements are false.

Here, the association set $\{\beta_j\}$ is defined as the probability of the events $\{\epsilon_j\}$ given all the measurements Ψ, $\beta_j = p\{\epsilon_j \Psi\}$. In this appendix, it is shown how β_j is related to the feature likelihood ratio of the measurement, e_j.

Let the set of validated measurements at time k be denoted as $\Psi_k = \{\psi_k(j)\}$, for $i = 1, \ldots, m_k$. In general, the following criteria must be satisfied:

$$(\psi - \bar{x})'(R + Q)^{-1}(\psi - \bar{x}) \leq \gamma \tag{A1}$$

where the γ is chi-squared distributed with n_z degrees of freedom. The association probability of β_j for a set of gated measurements can then be stated as $\beta_j = p\{\epsilon_j \Psi, m_k\}$. Using Bayes's rule, this can be expressed as

$$\beta_j = \frac{1}{c_1} p\{\Psi \epsilon_j, m_k\} p\{\epsilon_j m_k\} \tag{A2}$$

Assuming that the false measurements are uniformly distributed within the validation region, and the correct measurement location is assumed to be Gaussian with mean $\bar{x}$ and

covariance of $S = R + Q$, the probability density function for the correct measurement without the intensity feature is

$$\mathcal{L}_j = p\{\Psi\epsilon_j, m_k\} = P_G^{-1}\mathcal{N}(\psi_j\bar{x}, S) \tag{A3}$$

where P_G is the gating probability. Since the intensities are independent of location within the validation region, the probability density function of a single correct measurement, including the intensity feature, can be expressed as the product of the intensity likelihood ratio with $\mathcal{L}_j$ as follows:

$$p\{\psi\epsilon_j, m_k\} = P_1^{\tau}(f_j)\mathcal{L}_j \tag{A4}$$

and for incorrect measurements

$$p\{\psi\epsilon_j, m_k\} = P_0^{\tau}(f_j)V^{-1} \tag{A5}$$

where V is the volume of the validation region so for the all measurements, the probability density function is

$$p[\mathbf{\Psi}\epsilon_j, m_k] = P_1^{\tau}(f_j)[\prod_{j=1}^{m} p_0^{\tau}(f_j)]V^{-m+1}e_j \tag{A6}$$

for $j = 1, \ldots, m_k$ and

$$p[\mathbf{\Psi}\epsilon_j, m_k] = V^{-m}[\prod_{j=1}^{m} p_0^{\tau}(f_j)] \tag{A7}$$

for $j = 0$.

Using a nonparametric model considering

$$p[\epsilon_j m_k, x] = P_D P_G m^{-1} \tag{A8}$$

for $j = 1, \ldots, m_k$ and

$$p[\epsilon_j m_k, x] = 1 - P_D P_G \tag{A9}$$

for $j = 0$, the association probability β_j can be expressed as

$$\beta_j = P_1^{\tau}(f_j)[\prod_{j=1}^{m} p_0^{\tau}(f_j)]V^{-m+1}e_j P_D P_G m_k^{-1} \tag{A10}$$

for $j = 1, \ldots, m_k$ and

$$\beta_0 = V^{-m}[\prod_{j=1}^{m} p_0^{\tau}(f_j)](1 - P_D P_G) \tag{A11}$$

for $j = 0$.

Since the set of events $\{\epsilon_j\}$ are mutually exclusive and exhaustive,

$$\beta_0 + \sum_{j=1}^{m_k} \beta_j = 1 \tag{A12}$$

With some simplification, the overall results can be stated as

$$\begin{aligned} \beta_j &= p\{\epsilon_j\psi_j, m_k\} \\ &= \begin{cases} \frac{\mathcal{L}_i e_i}{1 - P_D P_G + \sum_{j=1}^{m_k} \mathcal{L}_j e_j}, \wedge j \neq 0 \\ \frac{1 - P_D P_G}{1 - P_D P_G + \sum_{j=1}^{m_k} \mathcal{L}_j e_j} \wedge j = 0 \end{cases} \end{aligned} \tag{A13}$$

Appendix B. IPDAF-Based Lane Tracking Algorithm (Detailed)

The overall IPDAF algorithm using the PDAF algorithm with special initialization and the termination steps is outlined in Algorithm A1 below.

Algorithm A1 IPDATracker.

▷ Inputs: Γ, Ψ
▷ Output: $\Gamma(Updated)$
▷ Section Variables : $\Lambda, \alpha, \alpha_T, \Omega$
▷ $\bar{x}$: Priors
▷ Ψ : Measurements
▷ Γ : Composite data structure for tracks
▷ Λ : A copy of Γ
▷ Ω : A set containing associated tracks
▷ α : A temporary variable
▷ α_T : A temporary set of tracks
$\Lambda = \Gamma.Tracks$
$\Omega \leftarrow \emptyset$
if $\Lambda.size() > 0$ **then**
  for `i=0; i<`$size(\Gamma)$`; i++` **do**
    $\hat{x}^i_{kk-1} \leftarrow$ Equation (34)
    $P^i_{kk-1} \leftarrow$ Equation (35)
    $W^i_k \leftarrow$ Equation (43)
    $\hat{z}_{kk-1} \leftarrow$ Equation (36)
    $S_k \leftarrow$ Equation (37)
    $V(k, \gamma) \leftarrow$ Equation (39)
    if $V.size() > 0$ **then**
      for $j = 0$ **to** V.size() **do**
        $L^i_k \leftarrow$ Equation (40)
        $e^i_k \leftarrow$ Equation (41)
      end for
      $\beta^i_k \leftarrow$ Equation (31)
      $\hat{x}^i_{kk} \leftarrow$ Equation (42)
      $P^i_{kk} \leftarrow$ Equation (44)
    else
      $\Lambda^i.x_k \leftarrow \hat{x}^i_k$
      $\Lambda^i.P_k \leftarrow P^i_k$
      $\Omega \leftarrow \Omega \cup 0$
      $\alpha \leftarrow$ `getTrackType`(Ω, Λ^i)
      $\Lambda^i.ID \leftarrow$ `getTrackID`(α, Λ^i)
      $\Lambda^i.type \leftarrow \alpha$
    end if
    $\Lambda^i.x_k \leftarrow \hat{x}^i_{kk}$
    $\Lambda^i.P_k \leftarrow P^i_{kk}$
    $\Omega \leftarrow \Omega \cup 1$
    $\alpha \leftarrow$ `getTrackType`(Ω, Λ^i)
    $\Lambda^i.ID \leftarrow$ `getTrackID`(α, Λ^i)
    $\Lambda^i.type \leftarrow \alpha$
  end for
  $\Gamma.NonAsscociated \leftarrow \Psi - \Omega$
  $\alpha_T \leftarrow$ `mergeTracks`(Λ)
  $\Gamma.Tracks \leftarrow$ `cleanTracks`(α_T)
else
  $\Gamma.NonAsscociated \leftarrow \Psi$
end if
return Γ

The algorithm assumes a composite data-type that is capable of capturing the evolution of the states over a period of time. As such, we use the `dot` notation to extract these properties. For instance, in our algorithm we use Γ as a variable that has all the information of all the tracks being considered in the problem. Various properties, such as *track type* and *track ID* (a unique identifier for each track), are extracted using the stated dot notation. Furthermore, the algorithm assumes the presence of the following auxiliary functions:

- `getTrackType()`: which returns the type of the track i, as temporary, retired or active;
- `getTrackID()`: which returns the unique identifier for the track;
- `mergeTracks()`: which fuses the supplied set of tracks; and
- `cleanTracks()`: removes dead tracks from the list.

References

1. Bila, C.; Sivrikaya, F.; Khan, M.A.; Albayrak, S. Vehicles of the Future: A Survey of Research on Safety Issues. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 1046–1065. [CrossRef]
2. Liu, Y.; Mukherjee, N.; Rajski, J.; Reddy, S.M.; Tyszer, J. Deterministic stellar bist for automotive ics. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2020**, *39*, 1699–1710. [CrossRef]
3. Kastner, R.; Michalke, T.; Adamy, J.; Fritsch, J.; Goerick, C. Task-Based Environment Interpretation and System Architecture for Next Generation ADAS. *IEEE Intell. Transp. Syst. Mag.* **2011**, *3*, 20–33. [CrossRef]
4. Wu, S.; Decker, S.; Chang, P.; Camus, T.; Eledath, J. Collision Sensing by Stereo Vision and Radar Sensor Fusion. *IEEE Trans. Intell. Transp. Syst.* **2009**, *10*, 606–614.
5. Gaikwad, V.; Lokhande, S. Lane Departure Identification for Advanced Driver Assistance. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 910–918. [CrossRef]
6. Song, W.; Yang, Y.; Fu, M.; Li, Y.; Wang, M. Lane Detection and Classification for Forward Collision Warning System Based on Stereo Vision. *IEEE Sens. J.* **2018**, *18*, 5151–5163. [CrossRef]
7. Dang, R.; Wang, J.; Li, S.E.; Li, K. Coordinated Adaptive Cruise Control System with Lane-Change Assistance. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2373–2383. [CrossRef]
8. Bevly, D.; Cao, X.; Gordon, M.; Ozbilgin, G.; Kari, D.; Nelson, B.; Woodruff, J.; Barth, M.; Murray, C.; Kurt, A.; Redmill, K.; Ozguner, U. Lane Change and Merge Maneuvers for Connected and Automated Vehicles: A Survey. *IEEE Trans. Intell. Veh.* **2016**, *1*, 105–120. [CrossRef]
9. Fletcher, L.; Apostoloff, N.; Petersson, L.; Zelinsky, A. Vision in and out of vehicles. *IEEE Intell. Syst.* **2003**, *18*, 12–17. [CrossRef]
10. Sanchez, N.; Alfonso, J.; Torres, J.; Menendez, J.M. ITS-based cooperative services development framework for improving safety of vulnerable road users. *IET Intell. Transp. Syst.* **2013**, *7*, 236–243. [CrossRef]
11. Borkar, A.; Hayes, M.; Smith, M.T. A Novel Lane Detection System With Efficient Ground Truth Generation. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 365–374. [CrossRef]
12. Dai, X.; Kummert, A.; Park, S.B.; Neisius, D. A warning algorithm for Lane Departure Warning system. In Proceedings of the 2009 IEEE Intelligent Vehicles Symposium, Xi'an, China, 3–5 June 2009; pp. 431–435.
13. Lin, B.; Lin, Y.; Fu, L.; Hsiao, P.; Chuang, L.; Huang, S.; Lo, M. Integrating Appearance and Edge Features for Sedan Vehicle Detection in the Blind-Spot Area. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 737–747.
14. Kim, Z. Robust Lane Detection and Tracking in Challenging Scenarios. *IEEE Trans. Intell. Transp. Syst.* **2008**, *9*, 16–26. [CrossRef]
15. Li, Q.; Chen, L.; Li, M.; Shaw, S.; Nüchter, A. A Sensor-Fusion Drivable-Region and Lane-Detection System for Autonomous Vehicle Navigation in Challenging Road Scenarios. *IEEE Trans. Veh. Technol.* **2014**, *63*, 540–555. [CrossRef]
16. Yuan, C.; Chen, H.; Liu, J.; Zhu, D.; Xu, Y. Robust Lane Detection for Complicated Road Environment Based on Normal Map. *IEEE Access* **2018**, *6*, 49679–49689. [CrossRef]
17. Lee, C.; Moon, J. Robust Lane Detection and Tracking for Real-Time Applications. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 4043–4048. [CrossRef]
18. Aly, M. Real time detection of lane markers in urban streets. In Proceedings of the 2008 IEEE Intelligent Vehicles Symposium, Eindhoven, The Netherlands, 4–6 June 2008; pp. 7–12.
19. Danescu, R.; Nedevschi, S. Probabilistic Lane Tracking in Difficult Road Scenarios Using Stereo-vision. *IEEE Trans. Intell. Transp. Syst.* **2009**, *10*, 272–282. [CrossRef]
20. Gopalan, R.; Hong, T.; Shneier, M.; Chellappa, R. A Learning Approach Towards Detection and Tracking of Lane Markings. *IEEE Trans. Intell. Transp. Syst.* **2012**, *13*, 1088–1098. [CrossRef]
21. Niu, J.; Lu, J.; Xu, M.; Lv, P.; Zhao, X. Robust Lane Detection using Two-stage Feature Extraction with Curve Fitting. *Pattern Recognit.* **2016**, *59*, 225–233. [CrossRef]
22. Bertozzi, M.; Broggi, A. GOLD: A parallel real-time stereo vision system for generic obstacle and lane detection. *IEEE Trans. Image Process.* **1998**, *7*, 62–81. [CrossRef]

23. Kuk, J.G.; An, J.H.; Ki, H.; Cho, N.I. Fast lane detection tracking based on Hough transform with reduced memory requirement. In Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems, Funchal, Portugal, 19–22 September 2010; pp. 1344–1349.
24. Yang, W.; Fang, B.; Tang, Y.Y. Fast and Accurate Vanishing Point Detection and Its Application in Inverse Perspective Mapping of Structured Road. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *48*, 755–766. [CrossRef]
25. Reichenbach, M.; Liebischer, L.; Vaas, S.; Fey, D. Comparison of Lane Detection Algorithms for ADAS Using Embedded Hardware Architectures. In Proceedings of the 2018 Conference on Design and Architectures for Signal and Image Processing (DASIP), Porto, Portugal, 10–12 October 2018; pp. 48–53.
26. Lee, Y.; Kim, H. Real-time lane detection and departure warning system on embedded platform. In Proceedings of the 2016 IEEE 6th International Conference on Consumer Electronics, Berlin, Germany, 5–7 September 2016; pp. 1–4.
27. Nieto, M.; Salgado, L.; Jaureguizar, F.; Cabrera, J. Stabilization of Inverse Perspective Mapping Images based on Robust Vanishing Point Estimation. In Proceedings of the IEEE Intelligent Vehicles Symposium, Istanbul, Turkey, 13–15 June 2007; pp. 13–15.
28. Su, H. Vanishing points in road recognition: A review. In Proceedings of the 2018 11th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 8–9 December 2018; Volume 2, pp. 132–136.
29. Yoo, J.H.; Lee, S.W.; Park, S.K.; Kim, D.H. A Robust Lane Detection Method Based on Vanishing Point Estimation Using the Relevance of Line Segments. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 3254–3266. [CrossRef]
30. Kong, H.; Audibert, J.Y.; Ponce, J. General Road Detection From a Single Image. *IEEE Trans. Image Process.* **2010**, *19*, 2211–2220. [CrossRef]
31. Wang, Y.; Teoh, E.K.; Shen, D. Lane detection and tracking using B-Snake. *Image Vis. Comput.* **2004**, *22*, 269–280. [CrossRef]
32. Andrade, D.C.; Bueno, F.; Franco, F.R.; Silva, R.A.; Neme, J.H.; Margraf, E.; Omoto, W.T.; Farinelli, F.A.; Tusset, A.M.; Okida, S.; et al. A Novel Strategy for Road Lane Detection and Tracking Based on a Vehicle's Forward Monocular Camera. *IEEE Trans. Intell. Transp. Syst. (Early Access)* **2018**, *20*, 1497–1507. [CrossRef]
33. Yue, M.; Hou, X.; Zhao, X.; Wu, X. Robust Tube-Based Model Predictive Control for Lane Change Maneuver of Tractor-Trailer Vehicles Based on a Polynomial Trajectory. *IEEE Trans. Syst. Man Cybern. Syst. (Early Access)* **2018**, *50*, 5180–5188. [CrossRef]
34. Zhu, B.; Yan, S.; Zhao, J.; Deng, W. Personalized Lane-Change Assistance System With Driver Behavior Identification. *IEEE Trans. Veh. Technol.* **2018**, *67*, 10293–10306. [CrossRef]
35. Coulombeau, P.; Laurgeau, C. Vehicle yaw, pitch, roll and 3D lane shape recovery by vision. In Proceedings of the IEEE Intelligent Vehicle Symposium, Versailles, France, 17–21 June 2002; pp. 619–625.
36. Feihu, Z.; Hauke, S.; Chao, C.; Christian, B.; Alois, K. A lane marking extraction approach based on Random Finite Set Statistics. In Proceedings of the Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 23–26 June 2013; pp. 1143–1148.
37. Ma, Y.; Havyarimana, V.; Bai, J.; Xiao, Z. Vision-Based Lane Detection and Lane-Marking Model Inference: A Three-Step Deep Learning Approach. In Proceedings of the 2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), Taipei, Taiwan, 26–28 December 2018; pp. 183–190.
38. Pan, X.; Shi, J.; Luo, P.; Wang, X.; Tang, X. Spatial As Deep: Spatial CNN for Traffic Scene Understanding. *arXiv* **2017**, arXiv:1712.06080.
39. Wang, Z.; Ren, W.; Qiu, Q. LaneNet: Real-Time Lane Detection Networks for Autonomous Driving. *arXiv* **2018**, arXiv:1807.01726.
40. Feng, J.; Wu, X.; Zhang, Y. Lane detection base on deep learning. In Proceedings of the 2018 11th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, China, 8–9 December 2018; Volume 1, pp. 315–318.
41. Yang, Z. Research on lane recognition algorithm based on deep learning. In Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 16–18 October 2019; pp. 387–391.
42. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
43. Brust, C.; Sickert, S.; Simon, M.; Rodner, E.; Denzler, J. Convolutional Patch Networks with Spatial Prior for Road Detection and Urban Scene Understanding. In Proceedings of the 10th International Conference on Computer Vision Theory and Applications—Volume 3: VISAPP, (VISIGRAPP 2015), Berlin, Germany, 11–14 March 2015; pp. 510–517.
44. Sharma, A.; Liu, X.; Yang, X.; Shi, D. A patch-based convolutional neural network for remote sensing image classification. *Neural Netw.* **2017**, *95*, 19–28. [CrossRef]
45. Kiryati, N.; Eldar, Y.; Bruckstein, A. A probabilistic Hough transform. *Pattern Recognit.* **1991**, *24*, 303–316. [CrossRef]
46. Bar-Shalom, Y.; Li, X.R.; Kirubarajan, T. *Estimation, Tracking and Navigation: Theory, Algorithms and Software*, 1st ed.; John Wiley and Sons: New York, NY, USA, 2001.
47. Bar-Shalom, Y.; Daum, F.; Huang, J. The probabilistic data association filter. *IEEE Control Syst.* **2009**, *29*, 82–100.
48. Gonzalez, R.C.; Woods, R.E. *Digital Image Processing*, 4th ed.; Addison-Wesley Longman Publishing Co., Inc.: Boston, MA, USA, 2017.
49. Duda, R.O.; Hart, P.E. Use of the Hough Transformation to Detect Lines and Curves in Pictures. *Commun. ACM* **1972**, *15*, 11–15. [CrossRef]
50. Galamhos, C.; Matas, J.; Kittler, J. Progressive probabilistic Hough transform for line detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition'99, Fort Collins, CO, USA, 23–25 June 1999; pp. 554–560.

51. Musicki, D.; Evans, R.; Stankovic, S. Integrated probabilistic data association. *IEEE Trans. Autom. Control* **1994**, *39*, 1237–1241. [CrossRef]
52. Huang, Y.; Chong, S.Y.; Song, T.L.; Lee, J.H. Multi-path data association for over-the-horizon radar using linear multitarget integrated probabilistic data association. In Proceedings of the 2018 International Conference on Control, Automation and Information Sciences (ICCAIS), Hangzhou, China, 24–27 October 2018; pp. 72–77.
53. Lerro, D.; Bar-Shalom, Y. Automated Tracking with Target Amplitude Information. In Proceedings of the 1990 American Control Conference, San Diego, CA, USA, 23–25 May 1990; pp. 2875–2880.
54. Mallick, M.; Scala, B. Comparison of Single-point and Two-point Difference Track Initiation Algorithms Using Position Measurements. *Acta Autom. Sin.* **2008**, *34*, 258–265. [CrossRef]
55. Hur, J.; Kang, S.N.; Seo, S.W. Multi-lane detection in urban driving environments using conditional random fields. In Proceedings of the 2013 IEEE Intelligent Vehicles Symposium (IV), Gold Coast, Australia, 23–26 June 2013; pp. 1297–1302.
56. Tusimple Benchmark. 2017. Available online: http://benchmark.tusimple.ai (accessed on 17 July 2019).